Examining the Rarity of Classes using Big Data

Sarbasri Sahoo¹, Abani Kumar Bisoi²

^{1,2} Department of Computer Science & Engineering, Einstein Academy of Technology & Management, Bhubaneswar, Odisha

Abstract

In the field of machine learning, a class imbalance occurs when there is a statistically significant difference in the number of instances (majority) between the two classes. Class imbalance can cause datasets to be biased toward the majority (negative) class in Machine Learning algorithms. This can have detrimental effects if false negatives are penalized more heavily than false positives. We investigate class rarity in big data through two case studies in our paper, each using a unique combination of three learners (logistic regression, random forest, and gradient-boosted trees) and three performance metrics (area under the precision-recall curve, geometric mean, and area under the receiver operating characteristic curve). Our trials produced class rarity, an exceptionally high degree of class imbalance.

Keywords: Big data, Class imbalance, Machine learning, Medicare fraud, POST Slowloris, Class rarity & under sampling.

I. Introduction

When called upon to define big data, researchers and practitioners in the field of data science frequently refer to the six V's: volume, variety, velocity, variability, value, and veracity [1]. Volume, most certainly the bestknown property of big data, is associated with the profusion of data produced by an organization. Variety covers the handling of structured, unstructured, and semi-structured data. Velocity takes into account how quickly data is manufactured, issued, and dealt with. Variability refers to the fluctuations in data. Value is often regarded as a critical attribute because it is required for effective decision-making. Veracity is associated with the fidelity of data. A definition of big data related to a minimum number of dataset instances has not been established in the literature. For example, in [2] this minimum was identified as 100,000 instances, but other works use 1,000,000 instances [3, 4]. The increasing reliance on big data applications is pushing the development of efficient knowledge-extractionmethods for this type of data.

Any dataset containing majority and minority classes, e.g., normal transactions and fraudulent transactions for a large bank over the course of a day, can be viewed as class- imbalanced. Various degrees of class imbalance exist, ranging from slightly imbalancedto rarity. Class rarity in a dataset is defined by comparatively inconsequential numbers of positive instances [5], e.g., the occurrence of 10 fraudulent transactions out of 1,000,000 total transactions generated daily for a bank. Binary classification is usually associated with class imbalance since many multi-class classification problems can be managed by breaking down the data into multiple binary classification tasks. The minority (positive) class, which comprises a smaller part of the dataset, is usually the class of interest in real-world problems [2], as opposed to the majority (negative) class, which comprises the larger part of the dataset. Although class imbalance affects both big and non-big data, the adverse effects are usually more perceptible in the former, due to the existence of extreme degrees of class imbalance within big data [6] as a result of voluminous over- representation of the negative (majority) class within datasets.

Machine Learning (ML) algorithms are usually better classifiers than traditional statistical techniques [7–9], but these algorithms cannot properly differentiate between majority and minority classes if the dataset is plagued by class rarity. The inability to sufficiently distinguish majority from minority classes is analogous to searching for the proverbial needle in a haystack and could result in the classifier labeling almost all instances as the majority (negative) class. Performance metric values based on such poor analysis would be deceptively high. When the occurrence of a false negative incurs a higher cost than a false positive, a classifier's bias towards the majority class may lead to adverse con-sequences [10].

Our work evaluates the classification performance of three learners (*Gradient-Boosted Trees*, *Logistic Regression, Random Forest*) with three performance metrics (*Area Under the Receiver Operating Characteristic (ROC) Curve, Area Under the Precision-Recall Curve, Geometric Mean*), and also compares results from two case studies involving imbalanced big data from different application domains. Class rarity was injected by randomly removing positive instances to artificially generate eight subsets of positive class instances (1,000, 750, 500, 400, 300, 200, 100, and 50). For comparative purposes, we also evaluated the

original positive class instances for both datasets. All model evalu- ations were performed through Cross-Validation. For our processing needs, we use the Apache Spark [11] and Apache Hadoop frameworks [12–14], both of which can handle big data.

The first case study is based on the Medicare Part B dataset [15, 16]. The original data- set contains 1,409 (0.038%) positive class instances out of 3,692,555 total instances. The results indicate that performance scores for the learners generally improve with the *Area*

Under the ROC Curve metric as the rarity level decreases, while corresponding scores with the Area Under the Precision-Recall Curve (AUPRC) and Geometric Mean (GM)metrics remain relatively unchanged. The second case study is based on a combination of two datasets: POST dataset [17] and Slowloris dataset [18

The remainder of this paper is organized as follows: Section "Related Work" provides an overview of literature that focuses on datasets with class rarity; Section "Case studies datasets" provides details on the Medicare Part B and POSTSlowloris Combined data- sets; Section "Methodologies" describes the different aspects of the methodology usedto develop and implement our approach, including injection of rarity, model evaluation, and experiment design; Section "Results and discussion" presents and discusses our empirical results; Section "Conclusion" concludes our paper with a summary of the work presented and suggestions for related future work.

II. Related work

Our search for related work found many more big data studies involving severe class imbalance than class rarity. It should be noted that research on class rarity is still in its infancy.

In [11], researchers examine *Evolutionary Under sampling* (EUS) in cases of class imbalance in big data, based on the initial knowledge that EUS had shown promise in addressing class imbalance in traditional data. The EUS approach is implemented within the Apache Spark framework, and compared with their previous implementation of EUS with the Apache Hadoop framework. The base learner in both implementations is the C4.5 decision tree learner which is incorporated into the overall class balancing and classification process. EUS provides a fitness function for a prototype selection method, where the fitness function aims to find the proper balance between reduction (under- sampling) of training data instances and classification performance [19].

An evaluation of the performance of several methods used to address class imbalance in big data was performed in [20], where all methods were implemented within theApache Hadoop framework, with RF as the base classifier. These methods included *Random Oversampling* (ROS), RUS, *Synthetic Minority Over-sampling Technique* (SMOTE), and a cost-sensitive learning version of RF. The datasets in this study ranged from approximately 435,000 to 5,700,000 instances, with feature set sizes between 2 and 41.Majority-to-minority class ratios varied between 80:20 and 77,670:1. There were several clear instances of big data datasets with class rarity in this work: (52 positive instances and 3,883,370 negative instances; 52 positive instances and 972,781 negative instances;

15 positive instances and 1,165,011 negative instances; 20 positive instances and 1,553,348 instances; 26 positive instances and 1,941,685 negative instances; and 52 positive instances and 3,883,370 negative ones. The results of the experiment were inconclusive, as there was no best model among these four diverse algorithms. The authors statethat the best performing algorithm depends on the number of mappers with Map Reducethat are chosen to run the experiment. For GM, the best overall values of ROS, RUS, SMOTE, and RF were 0.986, 0.984, 0.914, and 0.965, respectively. Just like the relatedwork in the previous paragraph, the focus of this related study is not on class rarity.

The work in [21] examined a modification to DeepQA, the technology that poweredIBM Watson on the Jeopardy! game show. DeepQA is a question-and-answer, naturallanguage processing system that can help professionals make critical and timely decisions [22]. The results show that regularized LR with over-sampling outperformed unregularized LR with over-sampling in terms of accuracy, which increased from 1.6 to28%. It is worth mentioning that for this study, some data scientists may not consider the total number of dataset instances (720,000) as big data.

Finally, in [5], the impact of class rarity on big data is evaluated. The researchers use publicly available Medicare data and map known fraudulent providers, from the *List of Excluded Individuals/Entities* (LEIE) [23], as labels for the positive class.

Case studies datasets

Our work includes two case studies. The dataset used in the first case study came from different application domain than the dataset used in the second case study. The first case study is based on the Medicare Part B dataset, which contains 1,409 (0.038%) positive class instances out of 3,692,555 total

instances. The second case study is based on a combination of two datasets: POST dataset and Slowloris dataset. The merged dataset, referred to as POSTSlowloris Combined, contains 6,646 (0.203%) positive instances out of 3,276,866 total instances.

Medicare Part B

The Medicare Physician and Other Supplier (Part B) dataset used in this paper spansyears 2012 to 2016. It is provided by the Centers for Medicare and Medicaid Services. The Part B dataset, which includes claims information for each procedure that a physician/provider performs in a given year, is derived from administrative claims data for Medicare beneficiaries enrolled in the Fee-For-Service program, where all claims information is recorded after payments are made [15]. Therefore, we can safely assume that these datasets are already reasonably cleansed.

POSTSlowloris

DDoS attacks are carried out through various methods designed to deny network avail- ability to legitimate users [24]. *Hypertext Transfer Protocol* (HTTP) contains several exploitable vulnerabilities and is often targeted for DDoS attacks [25, 26].

Data collection for the POST and Slowloris DDoS attacks was performed within a real-world network setting. An ad hoc Apache web server, which was set up within a campus network environment, served as a viable target. The Switchblade 4 tool from the Open Web Application Security Project and a Slowloris.py attack script [27] were used to generate attack packets for POST and Slowloris, respectively. Attacks were launchedfrom a single host computer in hourly intervals. Attack configuration settings, such asconnection intervals and number of parallel connections, were varied, but the same PHP form element on the web server was targeted during the attack.

III. Methodologies

This section is a report on the methodologies followed, including our reasons for choos- ing them. We discuss the big data processing framework, injection of rarity, one-hot encoding, classifiers, performance metrics, model evaluation, addressing of randomness, and experiment design.

Big data processing framework

To facilitate the use of ML in big data analytics, data engineers build algorithms within software modules or packages, ensuring that reliability, speed, and scale are factored in. For the ML tasks, we use a state-of-the-art library, *Machine Learning Library* (MLlib), provided by Apache Spark [28, 29], hereinafter referred to as Spark. Compared to tra- ditional ML methods, Spark is exponentially faster at data processing, and is one of the largest open source projects for big data processing [30]. In addition, we utilize Apache Hadoop [12–14], which provides *Hadoop Distributed File System* (HDFS), a scalable component capable of storing large files across node clusters, and also utilize *Yet AnotherResource Negotiator* (YARN) [31], a component used for job management and schedul- ing in *High Performance Computing* (HPC).

For performance stability, we kept our data partitions invariant and memory use fixedduring the experiments. Thus, the number of distributed data partitions and the num- ber of the cluster slave nodes were picked based on the available resources of our HPCcluster.

Injecting rarity

The Medicare Part B dataset contains 1,409 (0.038%) positive class instances out of 3,692,555 total instances, and the POSTSlowloris Combined dataset contains 6,646 (0.203%) positive instances out of 3,276,866 total instances.

From this original data, we artificially generate eight subsets with gradually decreas-ing numbers of positive class instances (1,000, 750, 500, 400, 300, 200, 100, and 50). To create each subset, we built the model with the same number of negative instances and randomly picked a basket of positive counts from the positive instances. Table 1, whichprovides information on the minority (positive) and majority (negative) classes, summa- rizes the datasets (subsets) used in our experiment.

Table 1 Summary of datasets								
Positives	Negatives	Total	%Positives	%Negatives				
a. Medicare Part I	В							
1,409 (all)	3,691,146	3,692,555	0.038	99.962				
1,000	3,691,146	3,692,146	0.027	99.973				
750	3,691,146	3,691,896	0.020	99.980				
500	3,691,146	3,691,646	0.014	99.986				

Examining the Rarity of Classes using Big Data

400	3,691,146	3,691,546	0.011	99.989
300	3,691,146	3,691,446	0.008	99.992
200	3,691,146	3,691,346	0.005	99.995
100	3,691,146	3,691,246	0.003	99.997
50	3,691,146	3,691,196	0.001	99.999
b. PostSlowloris (Combined			
6,646 (all)	3,270,220	3,276,866	0.203	99.797
1,000	3,270,220	3,271,220	0.031	99.969
750	3,270,220	3,270,970	0.023	99.977
500	3,270,220	3,270,720	0.015	99.985
400	3,270,220	3,270,620	0.012	99.988
300	3,270,220	3,270,520	0.009	99.991
200	3,270,220	3,270,420	0.006	99.994
100	3,270,220	3,270,320	0.003	99.997
50	3,270,220	3,270,270	0.002	99.998

One- hot encoding

Another factor needing attention is categorical features. In their raw form, these are generally not compatible with ML algorithms. Additionally, if the categorical features were indexed, some models assume that there is a logical order or a value of the indi- ces. Such subsets of categorical features are known as ordinal features. A nominal feature, unlike an ordinal one, is a categorical feature whose instances can take a value that cannot be organized in a logical sequence [7]. In our work, all categorical features (nominal) were transformed into dummy variables using a one-hot encoding method [32], allowing conversion of nominal features into numerical values. One disadvan-tage of this method is that a new number of features equaling C 1 is generated from each feature, where C is the number of categories belonging to the specific feature, and consequently, the total feature space increases in size.

Classifiers

Our work uses three off-the-shelf learners (RF, GBT, LR), all of which are available in the Spark MLlib. These classifiers were selected to provide good coverage of vari- ous ML model families. Performance-wise, the three classifiers are regarded favora- bly, and they incorporate both ensemble and non-ensemble algorithms, providing a reasonable breadth of fraud detection results for assessing the impact of rarity inBig Data [33, 34]. In this section, we describe each model and note configuration and hyperparameter changes that differ from the default settings.

• RF is a regression and classification model tflat employs an ensemble learning approacfl. RF constructs a large number of independent decision trees during training and returns a final model prediction tflat is tfle average or majority vote of tfle individual tree results. We build eacfl RF learner witfl 100 trees. The param- eter tflat cacfles node IDs for eacfl instance was set to true, and tfle maximum memory parameter was set to 1024 MB in order to minimize training time. The maximum bins parameter, wflicfl is for discretizing continuous features, was set to 2 since we use one-flot encoding on categorical variables.

Performance metrics

Accuracy or error rate is usually derived from a naive 0.50 threshold that is used in the prediction of one out of the two classes. This is usually impractical since in most real- world situations the two classes are imbalanced, creating the majority and minorityclass groups. The *Confusion Matrix* (CM) for a binary classification problem is shown in Table 2 [10], where positive, the class of interest, is the minority class and negative is the majority class.

• *True positive* (TP) are positive instances correctly identified as positive.

• *True negative* (TN) are negative instances correctly identified as negative.

• *False positive* (FP), also known as Type I error, are negative instances incorrectlyidentified as positive.

• *False negative* (FN), also known as Type II error, are positive instances incorrectlyidentified as negative.



From those four broad CM metrics, we may calculate other performance metrics that consider the rates between the positive and the negative class as follows.

Note that ROC curves are usually used when there are roughly equal numbers of instances for each class, in other words when the data is balanced [35]. On the otherhand, the use of Precision-Recall curves is preferred when there is a moderate to large class imbalance [35]. Since our datasets are severely class-imbalanced, the use of AUC in this study is for comparative purposes only.

Model evaluation

In ML, one of the commonly used model evaluation methods is *Cross-Validation* (CV), in which a portion of the data trains the model while the remaining portion validates the built model. Also known as rotation estimation, k-fold CV evaluates predictive models partitioning the original sample into several folds of approximately equal size. The inducer is trained and tested k times, where each time it is trained on k = 1 folds and tested on the remaining fold. This is to ensure that all data are used in the classification.

With imbalanced data, one typically uses stratified k-fold CV, where the minority and majority classes have representative proportions, in each fold, of the class labels from the training data. When compared to regular cross validation, the stratification scheme is generally better suited for addressing bias and variance [36].



Addressing randomness

Due to the random generation of the datasets (through random CV splits, the selection of random positive instances, and the random ordering of instances and features prior to model training), the datasets used to build and train the models may retain both goodand/or poor instances. Moreover, learners such as RF and GBT may randomly select instances during the construction of each tree. Such randomness may affect final modelperformance. To address this problem, we use a repetitive models strategy [37] in our work, with each model repeated 10 times.

Experiment design

The following procedure summarizes the algorithmic steps used in our proposed approach.

- 1. Distribute the data among HDFS.
- 2. Perform one-flot encoding: The categorical features space is indexed and one-flot encoded to dummy variables in order to exclude any assumed ordering, by the learner, between tfle categories of the nominal features. This produces new features equal to one less tflan tfle number of categories in eacfl feature.
- 3. Inject rarity by artificially generating eigflt subsets witfl gradually decreasing numbers of positive class instances (1,000, 750, 500, 400, 300, 200, 100, and 50). Addition- ally, we include tfle original datasets containing all instances of the positive class.
- 4. Perform 5-fold CV and randomly order tfle instances and feature space on all data- sets. Perform the 5-fold CV witfl eacfl of tfle tflree classifiers (RF, GBT, LR), and eval- uate eacfl model produced witfl tfle tflree performance metrics (AUC, AUPRC, GM).
- 5. Starting from step 3, repeat tfle entire process 10 times.

To sum things up, we assessed the performance of 2,700 models (2 datasets 5-fold CV 3learners \times 9 positive counts (rarity) \times 10 repetitions). In total, we obtained 8100 per-formance values for the three performance metrics involved.

IV. Results and discussion

Tables 3 and 4 show the average results for the Medicare Part B and POSTSlowloris case studies, respectively. Three performance metrics are used in this work (AUC, AUPRC, GM). The classification performance of three learners (GBT, LR, RF) is evaluated with each of the three metrics. The highest score for each row, where a row represents a given learner for a specified metric, is shown in italics.

PosCount	50	100	200	300	400	500	750	1000	All
a. AUC									
GBT	0.6906	0.7344	0.7649	0.7822	0.7862	0.7914	0.7888	0.7957	0.7945
LR	0.7368	0.7477	0.7737	0.7779	0.7946	0.7973	0.8006	0.7998	0.8057
RF	0.6059	0.6375	0.7214	0.7214	0.7369	0.7503	0.7614	0.7800	0.7962
b. GM									
GBT	0.0063	0	0.0032	0.0052	0.0045	0.0020	0.0033	0.0042	0.0112
LR	0	0	0.0032	0.0077	0.0067	0.0080	0.0065	0.0156	0.0153
RF	0	0	0	0	0	0	0	0	0.0012
c. AUPRC									
GBT	0.0006	0.0003	0.0018	0.0020	0.0019	0.0027	0.0035	0.0046	0.0055
LR	0.0001	0.0002	0.0005	0.0013	0.0008	0.0017	0.0019	0.0025	0.0031
RF	0.0002	0.0001	0.0011	0.0010	0.0011	0.0013	0.0016	0.0025	0.0037

Table 3 Medicare Part B average results

PosCount	50	100	200	300	400	500	750	1000	All
a. AUC									
GBT	0.9948	0.9912	0.9963	0.9951	0.9903	0.9965	0.9979	0.9979	0.9990
LR	0.9903	0.9952	0.9915	0.9907	0.9841	0.9888	0.9909	0.9888	0.9877
RF	0.9809	0.9824	0.9818	0.9762	0.9764	0.9769	0.9766	0.9775	0.9753
b. GM									
GBT	0.6951	0.7422	0.7769	0.7723	0.7949	0.7745	0.7766	0.7484	0.7883

Examining the Rarity of Classes using Big Data

GBT 0.5465 0.6266 0.6572 0.6887 0.7363 0.7202 0.7561 0.760									
LR 0.0874 0.2199 0.3316 0.4235 0.4738 0.5134 0.5791 0.584	GBT 0	0.5465 0.62	0.6572	0.6887	0.7363	0.7202	0.7561	0.7605	0.9197
	LR 0	0.0874 0.21	99 0.3316	0.4235	0.4738	0.5134	0.5791	0.5842	0.7942

Generally speaking, from Table 3, part a (AUC), we observe that the performance increases when the number of PosCounts increases for all three learners, and "all" posi- tives reports the best performance. For parts b (GM), and c (AUPRC), we can see thatthe results are almost zero for all PosCounts. Particularly noticeable are the distinct 0 values for RF for all PosCounts in part b. This means that RF failed to correctly classify any positive counts, thus resulting in a TP_{rate} value of 0. Consequently, the GM score, which relies on the product of TP_{rate} and TN_{rate}, is also 0. Note that RF with 100 trees builds 100 Decision Trees separately and then takes majority voting. Therefore, if inevery instance prediction, the majority (more than 50) trees decided incorrectly, the final classification for RF will also be incorrect. We observe, for the most part, that the performance increases when the number of PosCounts increases for all three learners, and "all" positives reports the best performance. Furthermore, GBT reports the best performance, followed by LR and RF.It is very noticeable that GBT performs reasonably well with even a PosCounts of 50, compared to LR and RF.

1000000000000000000000000000000000000

Table 6 Tukey's HSD Test (PosCounts)										
Case study	Learner	GBT	GBT		LR			RF		
	Sampling	AUC	AUPR	GM	AUC	AUPR	GM	AUC	AUPR	GM
a. Medicare Part B	50	d	d	-	e	e	-	f	d	-
	100	с	d	-	de	e	-	e	d	-
	200	b	cd	-	cd	de	-	d	cd	-
	300	ab	cd	-	bc	cd	-	d	cd	-
	400	ab	cd	-	abc	cde	-	cd	cd	-
	500	a	bc	-	abc	bc	-	с	cd	-
	750	ab	abc	-	ab	bc	-	bc	bc	-
	1000	a	ab	-	abc	ab	-	ab	b	-
	All	а	а	-	а	а	-	а	а	-
b. POSTSlowloris Combined	50	abc	f	d	abc	h	f	abc	g	-
	100	bc	e	c	а	g	f	a	f	-
	200	ab	de	ab	ab	f	e	ab	e	-
	300	abc	cd	abc	abc	e	d	cd	d	-
	400	с	b	а	с	d	с	cd	c	-
	500	ab	bc	abc	abc	c	ab	bcd	c	-
	750	а	b	ab	ab	b	ab	cd	b	-
	1000	a	b	bc	abc	b	bc	abcd	b	-
	All	a	a	a	bc	a	a	d	a	-

Table 7	Tukey's HSD Test (Learners)

Case study	Learner	Metrics			
		AUC	AUPRC	GM	
Medicare Part B	GBT	b	a	a	
	LR	а	b	a	
	RF	с	b	b	
POSTSlowloris Combined	GBT	а	а	а	
	LR	b	b	b	
	RF	с	с	с	

Combined case study, a noticeable deterioration of performance occurs at PosCountsof 100 and below. As a result, we infer that PosCounts of 200 and below, and 100 and below, are solid indicators of class rarity for the first and second case study, respectively.

V. Conclusion

We employ three learners (GBT, LR, RF) and three performance metrics (AUC, GM, AUPRC) to uniquely investigate class rarity in big data. Through our comparative analysis, we demonstrate the effectiveness and versatility of our method with two case studies involving imbalanced big data from different application domains.

For the Medicare Part B case study, we observe that classification performance scores for the learners generally improve for the AUC metric as the number of PosCounts increases, with "all" positives reporting the best performance. The other metrics have scores of zero or approximately zero. With regard to the POSTSlowloris Combined case study, the AUC metric yields very high performance results for the learners, while coresponding scores for the other metrics are, for the most part, noticeably lower.

References

- Katal A, Wazid M, Goudar R. Big data: issues, challenges, tools and good practices. In: 2013 Sixth International Con-ference on contemporary computing (IC3). NewYork: IEEE; 2013. p. 404–409.
- [2]. Leevy JL, Khoshgoftaar TM, Bauder RA, Seliya N. A survey on addressing high-class imbalance in big data. J Big Data. 2018;5(1):42.
- [3]. Soltysik RC, Yarnold PR. Megaoda large sample and big data time trials: separating the chaff. Optim Data Anal.2013;2:194-7.
- [4]. Cao M, Chychyla R, Stewart T. Big data analytics in financial statement audits. Account Horizons. 2015;29(2):423-9.
- [5]. Bauder RA, Khoshgoftaar TM, Hasanin, T. An empirical study on class rarity in big data. In: 2018 17th IEEE Interna-tional Conference on machine learning and applications (ICMLA). Newyork: IEEE ; 2018. p. 785–790. IEEE
- [6]. Bauder R, Khoshgoftaar T. Medicare fraud detection using random forest with class imbalanced big data. In: 2018IEEE International Conference on information reuse and integration (IRI). Newyork: IEEE; 2018. p. 80–87.
- [7]. Witten IH, Frank E, Hall MA, Pal CJ. Data mining: practical machine learning tools and techniques. Amsterdam: Mor-gan Kaufmann; 2016.
- [8]. Olden JD, Lawler JJ, Poff NL. Machine learning methods without tears: a primer for ecologists. Q Rev Biol.2008;83(2):171–93.
- [9]. Galindo J, Tamayo P. Credit risk assessment using statistical and machine learning: basic methodology and riskmodeling applications. Comput Econ. 2000;15(1):107-43.
- [10]. Seliya N, Khoshgoftaar TM, Van Hulse J. A study on the relationships of classifier performance metrics. In: 2009 21stIEEE International Conference on tools with artificial intelligence. Newyork: IEEE; 2009. p. 59–66.
- [11]. Triguero I, Galar M, Merino D, Maillo J, Bustince H. Herrera, F. Evolutionary undersampling for extremely imbalancedbig data classification under apache spark. In: Evolutionary Computation (CEC), 2016 IEEE Congress on; Newyork: IEE; 2016. p. 640-647.
- [12]. Apache Hadoop. http://hadoop.apache.org/
- [13]. Venner J. Pro Hadoop. Berkeley: Apress; 2009.
- [14]. White T. Hadoop: the definitive guide. Sebastopol: O'Reilly Media Inc; 2012.
- [15]. Bauder RA, Khoshgoftaar TM, Hasanin T. Data sampling approaches with severely imbalanced big data for medicarefraud detection. In: 2018 IEEE 30th International Conference on tools with artificial intelligence (ICTAI). Newyork: IEEE; 2018. p. 137–142.
- [16]. Johnson JM, Khoshgoftaar TM. Medicare fraud detection using neural networks. J Big Data. 2019;6(1):63.
- [17]. Calvert C, Khoshgoftaar TM, Kemp C, Najafabadi MM. Detecting slow http post dos attacks using netflow features. In: The Thirty-second International FLAIRS Conference (2019).
- [18]. Calvert C, Khoshgoftaar TM, Kemp C, Najafabadi MM. Detection of slowloris attacks using netflow traffic. In: 24thISSAT International Conference on reliability and quality in design. 2018; p. 191–196.
- [19]. García S, Herrera F. Evolutionary undersampling for classification with imbalanced datasets: proposals and tax-onomy. Evol Comput. 2009;17(3):275–306.
- [20]. Del Río S, López V, Benítez JM, Herrera F. On the use of mapreduce for imbalanced big data using random forest. InfSci. 2014;285:112–37.
- [21]. Baughman AK, Chuang W, Dixon KR, Benz Z, Basilico J. Deepqa jeopardy! gamification: a machine-learning perspec-tive. IEEE Trans Comput Intell AI Games. 2013;6(1):55–66.
- [22]. Ferrucci D, Brown E, Chu-Carroll J, Fan J, Gondek D, Kalyanpur AA, Lally A, Murdock JW, Nyberg E, Prager J, et al. Building watson: an overview of the deepqa project. AI Mag. 2010;31(3):59–79.
- [23]. LEIE: Medicare provider utilization and payment data: physician and other supplier. https://oig.hhs.gov/exclusions/index.asp
- [24]. Liu Y-h, Zhang H-q, Yang Y-j. A dos attack situation assessment method based on qos. In: Proceedings of 2011 Inter-national Conference on computer science and network technology. Newyork: IEEE; 2011. p. 1041–1045.
- [25]. Yevsieieva O, Helalat SM. Analysis of the impact of the slow http dos and ddos attacks on the cloud environment. In: 2017 4th International scientific-practical Conference problems of infocommunications. Science and technology(PIC S&T). Newyork: IEEE; 2017. p. 519–523.
- [26]. Hirakaw T, Ogura K, Bista BB, Takata T. A defense method against distributed slow http dos attack. In: 2016 19thInternational Conference on networkbased information systems (NBiS). Newyork: IEEE; 2016. p. 519–523.
- [27]. Slowloris.py. https://github.com/gkbrk/slowloris
- [28]. Apache Spark MLlib. https://spark.apache.org/mllib/
- [29]. Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: cluster computing with working sets. HotCloud.2010;10:95.
- [30]. Meng X, Bradley J, Yuvaz B, Sparks E, Venkataraman S, Liu D, Freeman J, Tsai D, Amde M, Owen S, et al. Mllib: Machinelearning in apache spark. JMLR. 2016;17(34):1–7.
- [31]. Vavilapalli VK, Murthy AC, Douglas C, Agarwal S, Konar M, Evans R, Graves T, Lowe J, Shah H, Seth S, et al. Apache hadoop yarn: yet another resource negotiator. In: Proceedings of the 4th Annual Symposium on cloud computing.Newyork: ACM; 2013. p. 5.
- [32]. Herland M, Khoshgoftaar TM, Bauder RA. Big data fraud detection using multiple medicare data sources. J Big Data. 2018;5(1):29
- [33]. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A. Mining data with rare events: a case study. In: 19th IEEE International Conference on tools with artificial intelligence (ICTAI 2007). Newyork: IEEE; 2007. vol 2, p. 132–139. IEEE.
- [34]. Herland M, Khoshgoftaar TM, Wald R. A review of data mining using big data in health informatics. J Big data.2014;1(1):2.
- [35]. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the roc plot when evaluating binary classi-fiers on imbalanced datasets.

Examining the Rarity of Classes using Big Data

PLoS ONE. 2015;10:0118432.

Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th International Joint [36].

Conference on artificial intelligence. Burlington: Morgan Kaufmann Publishers Inc; 1995.Vol 2, p. 1137–1143. Van Hulse J, Khoshgoftaar TM, Napolitano A. An empirical comparison of repetitive undersampling techniques. In: 2009 IEEE International Conference on information reuse & integration. Newyork: IEEE; 2009. p. 29–34. [37].

Gelman A. Analysis of variance-why it is more important than ever1. Ann Stat. 2005;33(1):1-53. [38].